

Micro C Buddy System 用户操作手册

发布日：2005 年 5 月 1 日

北京北阳电子技术有限公司.
北京市海淀区上地信息产业基地中黎科技园 1 号楼 6 层
电话: 86-10-62981668 传真 ：86-10-62985972 邮编：100085

版权声明

版本说明

2003.5.22　V1.0　史小平　　初始版本

目录

用户操作手册

1 引言

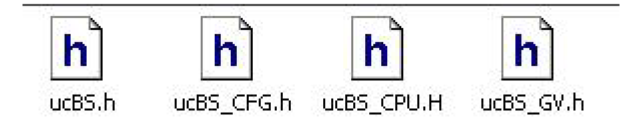
Micro C Buddy System 简称 ucBS，是一个动态管理内存的系统。顾名思义，Micro C Buddy System 全部采用 C 语言实现，C 语言具有与硬件平台无关的特性，所以 Micro C Buddy System 可以在任何支持C语言的CPU上运行。目前我们已经移植到了两类CPU上，分别是unsp 和 MCS8051，后续根据具体的应用需求，会慢慢增加其他 CPU 平台的移植版本，因为移植工作是非常简单。

2 版本命名规范

CPU	库文件名	备注
spl161001	ucBSAx.x.x.lib	x.x.x表示版本号
spl162001	ucBSBx.x.x.lib	x.x.x表示版本号
8051	ucBSCx.x.x.lib	x.x.x 表示版本号

3 头文件说明

发布的 ucBS 除了库文件名之外，还有一些头文件，见下图



头文件的意义如下：
ucBS.h ：包含了给用户提供 API 的接口声明
ucBS_CFG.h：ucBS 的配置文件
ucBS_CPU.h：根据不同 CPU 定义了一些抽象数据类型
ucBS_GV.h： ucBS 定义的全局变量。

4 配置文件

配置的选项定义如下：

```
#define BS_MIN_BLOCK 8 //Min Block Size
```

BS_MIN_BLOCK 定义 ucBS 管理的最小块大小。如果用户申请比最小块还小的空间，系统仍然会给用户 BS_MIN_BLOCK 大小的空间。由于 Buddy System 的特性，只能定义为 2 的幂数值。

#define BS_MAX_BLOCK 1024 //Max Block Size

BS_MAX_BLOCK 定义 ucBSucBS 管理的最大块大小，如果用户要申请比最大块还大的空间，一定会失败。由于 Buddy System 的特性，只能定义为 2 的幂数值。

#define BS_NUM 8

BS_NUM 定义了 ucBS 管理各种块的类型的总数。比如定义 BS_MIN_BLOCK 为 8，BS_MAX_BLOCK 为 1024，则有以下几种类型的块：

8 16 32 64 128 256 512 1024

共 8 种，所以 BS_NUM 必须定义为 8。用户自己配置的时候，必须根据 BS_MIN_BLOCK 和 BS_MAX_BLOCK 的值来确定 BS_NUM 的值。

#define BS_STACK_RESERVE_SIZE 1024 //Stack Reserve Size

BS_STACK_RESERVE_SIZE 定义预留空间的大小。比如系统交给 ucBS 管理的空间为 4K,如果预留 1k，那么 ucBS 管理的就是前面的 3K，一般的系统中定义为 0。

#define BS_LIST_MERGE_SIZE 4 //When the list up to this size ,merge the list

BS_LIST_MERGE_SIZE 定义当相同块的个数达到 BS_LIST_MERGE_SIZE，考虑合并。一般定义为 4．由于 ucBS 的特性，只能定义为 2 的倍数．

#define BS_MEM_RECORD_SIZE 4 //16/4 =4

BS_MEM_RECORD_SIZE 在移植 ucBS 时由移植人员定义，对于上层用户来说没有任何意义．

#define BS_MEM_SIZE 26*1024 + 256 + 128 + 16 + 845

BS_MEM_SIZE 定义交给 ucBS 管理的总空间的大小．对于 unsp 来说，可以根据 linker 来决定，所以没有这一配置．

5 用户接口

BSINTU ucBSInit(void);

功能描述：初始化 ucBS

入口参数：void

返回参数: 成功: BS_NO_ERR

不成功:非 BS_NO_ERR，

BSPTR ucBSMalloc(BSINTU size);

功能描述： 申请内存

入口参数：size 表示要申请内存块的大小

返回参数： 成功：有效的存地址

不成功 ：BS_NULL

void ucBSFree(BSPTR ptr);

功能描述： 释放内存

入口参数：释放内存的地址

返回参数：无

注：ucBS 在内部作了错误处理，如果 ucBS 能够识别出要释放的内存地址不可能是有效地址，系统会进入 While(1)死循环.

BSINTU ucBS_GetTotalMem();

功能描述：获取 ucBS 剩余空间的总数

入口参数：无

返回参数：剩余空间总数.

注：原计划这个函数是不公开给用户的，所以中间带了下划线，后来考虑到它在测试当中还是非常有用的，所以公开了.

6 unSP 的特殊接口

unSP 由于 CPU 的特性，ucBS 还提供对片外的内存的管理. 其配置方法和 API 的名字的使用同上面提到的类似，只是在名字后添加了 Ext.

如：

```
BSINT32U ucBSExtInit(void);
BSINT32U ucBSExtMalloc(BSINT32U  size);
void     ucBSExtFree(BSINT32U ptr);
BSINT32U ucBSExt_GetTotalMem();
```

7 使用步骤

1) 先设定配置文件
2) 在一个.C 文件(一般是 mainc 文件)中定义宏 #define BS_CREATE_VAR
3) 在该 C 文件中包含 ucBS.h 文件. #include "ucBS.h"
4) 在需要使用 ucBS 的地方，只需要包含 ucBS.h 文件就行了.

8 Demo 程序

以下代码完全取自一个文件，用户可以把它拷贝到一个 main.c 文件，建立工程运行.

```
#define CREATE_BS_VAR
//如果没有设置片外，就不需要定义下面的宏了
#define CREATE_BSEXT_VAR

#include "ucBS.h"
int main()
{
        void* ptr;
        unsigned long extptr;
        unsigned int total1;
        unsigned int total2;

        unsigned long total_ext1;
        unsigned long total_ext2;

        //Init
        ucBSInit();
        ucBSExtInit();

        total1          = ucBS_GetTotalMem();
        total_ext1 = ucBSExt_GetTotalMem();

        //malloc and free
        ptr             = ucBSMalloc(8);
        ucBSFree(ptr);

        //check size
        total2 = ucBS_GetTotalMem();
        if(total2 != total1)
                while(1);

        //malloc and free
        extptr = ucBSExtMalloc(129);
        ucBSExtFree(extptr);

        //check size
        total_ext2 = ucBSExt_GetTotalMem();
        if(total_ext2 != total_ext1)
                while(1);

        return 0;
}
```

9 与 OS 的结合

ucBS 提供回调函数来支持与 OS 的整合，库文件中的回调函数默认是空的，所以默认是不支持 OS 的．如果要于 OS 整合，用于需要重新实现回调函数．

与 MiniOS 的结合例子如下：

```
void ucBSMalloc_EnterHook()
{
    unOSIrqOff();
}
void ucBSMalloc_ExitHook()
{
    unOSIrqOn();
}
void ucBSFree_EnterHook()
{
    unOSIrqOff();
}
void ucBSFree_ExitHook()
{
    unOSIrqOn();
}
```